Slide 1

Lecture 3

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

**Internal Memory
Cache**

From Computer Desktop Encyclopedia
© 1999 The Computer Language Co. Inc.

<u>Ch 4, Ch 5 [Sta10]</u>
Key Characteristics
Locality
Cache
Main Memory

Slide 2

Key Characterics of Memories / Storage

Location	Performance
Processor	Access time
Internal (main)	Cycle time
External (secondary)	Transfer rate
Capacity	**Physical Type**
Word size	Semiconductor
Number of words	Magnetic
Unit of Transfer	Optical
Word	Magneto-Optical
Block	**Physical Characteristics**
Access Method	Volatile/nonvolatile
Sequential	Erasable/nonerasable
Direct	**Organization**
Random	
Associative	

(Sta10 Table 4.1)

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   2

Slide 3

Goals

- I want my memory lightning fast
- I want my memory to be gigantic in size

- Register access viewpoint
  - data access as fast as HW register
  - **data size as large as memory**

    cache

    HW solution

- Memory access viewpoint
  - data access as fast as memory
  - **data size as large as disk**

    virtual memory

    HW help for SW solution

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   3

Slide 4

Memory Hierarchy

- Most often needed data kept close
- Access to small data sets can be made fast
  - simpler circuits
  - smaller gate delays
- Faster ~ more expensive
- Large can be bigger and cheaper (per byte)

(Sta10 Fig 4.1)

up: smaller, faster, more expensive, more frequent access
down: bigger, slower, less expensive, less frequent access

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   4

Slide 5

Principle of locality (*paikallisuus*)

- In any given time period, memory references occur only to a <u>small subset</u> of the whole address space
= The reason why memory hierarchies work

Prob (small data set) = 95%   "Cost" (small data set) = 0.01 μs
Prob (the rest) = 5%   "Cost" (the rest) = 0.1 μs

Aver cost = 95% * 0.01 μs + 5% * 0.1 μs = 0.015 μs

- Average cost is close to the cost of small data set
- How to determine data for that small set?
- How to keep track of it?

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   5

Slide 6

Principle of locality

- In any given time period
  - Memory references occur only to a <u>small subset</u> of the whole address space
- Temporal locality (*ajallinen*)
  - It is likely that a data item referenced a <u>short time ago</u> will be referenced <u>again</u> soon
- Spatial locality (*alueellinen*)
  - It is likely that a data items <u>close</u> to the one referenced a short time ago will be referenced soon

MEM: | 345 | 23 | 71 | 8 | 305 | 63 | 91 | 2 |

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   6
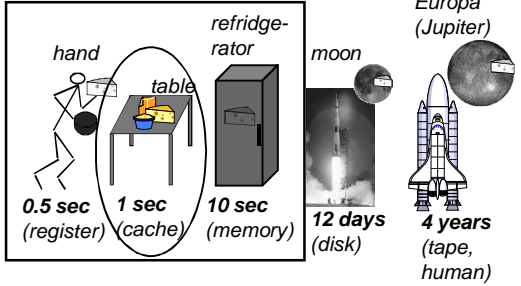
Computer Organization II

Cache

Computer Organization II, Autumn 2010, Teemu Kerola 5.11.2010 7

Teemu's Cheesecake

Register, on-chip cache, memory, disk, and tape speeds relative to times locating cheese for the cheese cake you are baking...
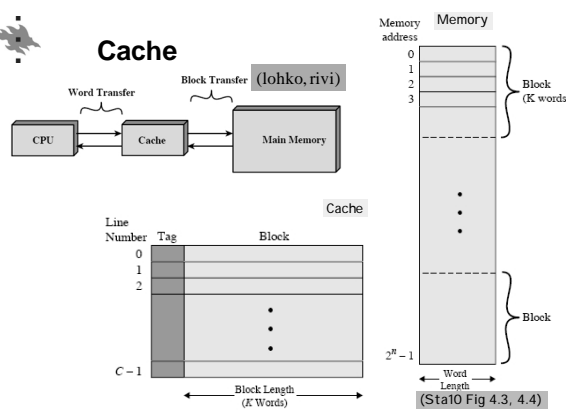

hand	table / refrigerator	moon	Europa (Jupiter)
0.5 sec (register)	1 sec (cache) / 10 sec (memory)	12 days (disk)	4 years (tape, human)

Computer Organization II, Autumn 2010, Teemu Kerola 5.11.2010 8

Cache Memory (välimuisti)

- How to access main memory as fast as registers?
- Locality → Use (CPU) cache!
  - Keep most probably referenced data in fast cache close to processor, and rest in memory
  - Most of data accesses only to cache
    - hit ratio 0.9-0.99
  - Cache is much smaller than main memory
  - Cache is (much) more expensive (per byte) than memory
- Note: file cache is another thing...

Computer Organization II, Autumn 2010, Teemu Kerola 5.11.2010 9

Cache

(Sta10 Fig 4.3, 4.4)

Computer Organization II, Autumn 2010, Teemu Kerola 5.11.2010 10

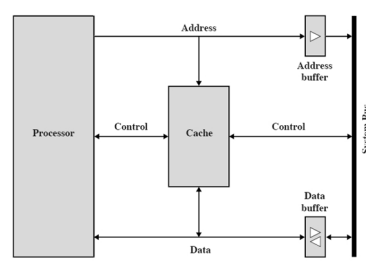## Cache Read

(RA = Real Address, Main memory address)

(Sta10 Fig 4.5)

Computer Organization II, Autumn 2010, Teemu Kerola 5.11.2010 11

Cache Organization

(Sta10 Fig 4.6)

Computer Organization II, Autumn 2010, Teemu Kerola 5.11.2010 12

Cache Design

Cache Size	Write Policy
Mapping Function	Write through
Direct	Write back
Associative	Write once
Set Associative	**Line Size**
Replacement Algorithm	**Number of caches**
Least recently used (LRU)	Single or two level
First in first out (FIFO)	Unified or split
Least frequently used (LFU)	
Random	(Sta10 Table 4.1)

- Cache Size & Line Size
  - Many blocks help for temporal locality
  - Large blocks help for spatial locality
  - Larger cache is slower
  - Multi-level cache

Typical sizes:
L1: 8 KB - 64 KB
L2: 256 KB - 8 MB
L3: 2 MB - 48 MB

Discussion?

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   13

Mapping

- Which block (if any) contains the referenced memory location?
  - Is the block in cache?
  - Where in the cache is it located?

- Solutions
  - Direct mapping (*suora kuvaus*)
    - One possible location
  - Fully associative mapping (*täysin assosiatiivinen*)
    - Any possible location
  - Set associative mapping (*joukkoassosiatiivinen*)
    - Some possible locations

Cache simulation tools:
http://www.ecs.umass.edu/ece/koren/architecture/Cache/frame0.htm

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   14

Direct Mapping

- Each block has only one possible location (line) in cache
  - determined by index bits
- Several blocks may map into same cache line
  - identified with different tag bits

0x2480
0x6480
0xA480

Block number (in memory)

34 bit address (byte address)

tag	index	byte offset

21   8   5

Cache line size = Block size = 2^5 = 32 B

Unique bits that are different for each block, Stored into cache line

Fixed location in cache → fixed cache size = 2^8 = 256 blocks = 8 KB

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   15

Direct Mapping Example

Word = 4B (here)

Block size = 2^3 = 8 bytes = 64 bits
Cache line size

ReadW I2, 0xA4

8 bit address (byte address)

tag index offset
2   3   3
10   100   100

No match

compare

	tag	block, 64b
000:		
001:		
010:		
011:	01	54 A7 00 91 23 66 32 11
100:	11	77 55 55 66 66 22 44 22
101:	01	65 43 21 98 76 65 43 32
110:		

Read new memory block from memory address 0xA0=1010 0000 to cache location 100, update tag, and then continue with data access

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   16

Direct Mapping Example 2

ReadW I2, 0xB4

tag index offset
2   3   3
10   110   100

compare

Match

start with 4th byte

	tag	block 64
000:		
001:		
010:		
011:	01	54 A7 00 91 23 66 32 11
100:	11	77 55 55 66 66 22 44 22
101:	01	65 43 21 98 76 65 43 32
110:	10	00 11 22 33 44 55 66 77
111:		

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   17

Fully Associative Mapping (6)

- Each block can be in any cache line
  - tag must be complete block number

Alpha AXP uses 34 bit memory addresses
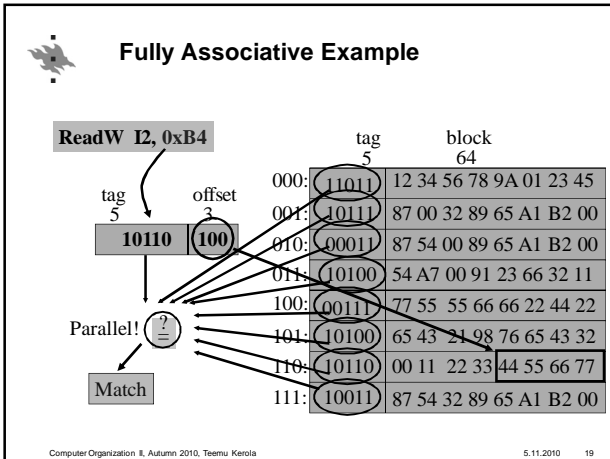
Block number (in memory)

Offset from the beginning of the block (in bytes)

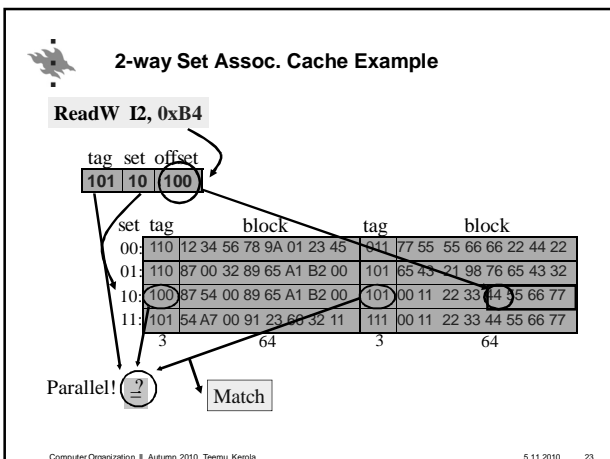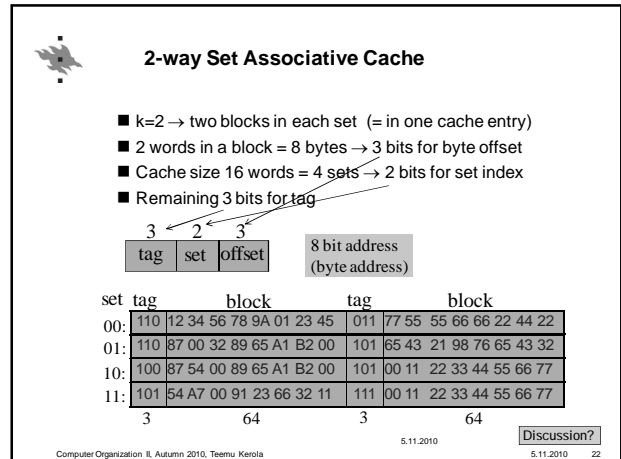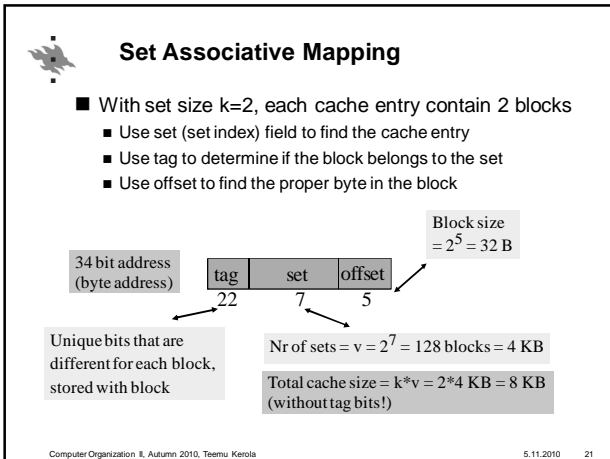Block size = 2^5 = 32 B

34 bit address (byte address)

tag	offset

29   5

Unique bits that are different for each block

Each block can be anywhere Cache size can be any number of blocks

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   18

Fully Associative Example

ReadW I2, 0xB4

tag   block
5   64

tag   offset
5   3

10110   100

Parallel!   ?=

Match

	tag	block
000:	10111	12 34 56 78 9A 01 23 45
001:	10111	87 00 32 89 65 A1 B2 00
010:	00011	87 54 00 89 65 A1 B2 00
011:	10100	54 A7 00 91 23 66 32 11
100:	00111	77 55 55 66 66 22 44 22
101:	10100	65 43 21 98 76 65 43 32
110:	10110	00 11 22 33 44 55 66 77
111:	10011	87 54 32 89 65 A1 B2 00

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   19

Fully Associative Mapping

- Lots of circuits
  - tag fields are long - wasted space?
  - each cache line tag must be compared <u>in parallel</u> with the memory address tag
    - lots of wires, comparison circuits
    - large surface area on chip
- Final comparison "or" has large gate delay
  - did any of these 64 comparisons match?
    - log2(64) = 6 levels of binary OR-gates
  - how about 262144 comparisons?
    - 18 levels?
- ⇒ Can use it only for small caches

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   20

Set Associative Mapping

- With set size k=2, each cache entry contain 2 blocks
  - Use set (set index) field to find the cache entry
  - Use tag to determine if the block belongs to the set
  - Use offset to find the proper byte in the block

Block size
$= 2^5 = 32$ B

34 bit address
(byte address)

tag	set	offset
22	7	5

Unique bits that are different for each block, stored with block

Nr of sets = $v = 2^7 = 128$ blocks = 4 KB

Total cache size = $k*v = 2*4$ KB = 8 KB (without tag bits!)

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   21

2-way Set Associative Cache

- k=2 → two blocks in each set (= in one cache entry)
- 2 words in a block = 8 bytes → 3 bits for byte offset
- Cache size 16 words = 4 sets → 2 bits for set index
- Remaining 3 bits for tag

3   2   3

tag	set	offset

8 bit address
(byte address)

set	tag	block	tag	block
00:	110	12 34 56 78 9A 01 23 45	011	77 55 55 66 66 22 44 22
01:	110	87 00 32 89 65 A1 B2 00	101	65 43 21 98 76 65 43 32
10:	100	87 54 00 89 65 A1 B2 00	101	00 11 22 33 44 55 66 77
11:	101	54 A7 00 91 23 66 32 11	111	00 11 22 33 44 55 66 77
	3	64	3	64

5.11.2010

Discussion?

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   22

2-way Set Assoc. Cache Example

ReadW I2, 0xB4

tag set offset

101	10	100

set	tag	block	tag	block
00:	110	12 34 56 78 9A 01 23 45	011	77 55 55 66 66 22 44 22
01:	110	87 00 32 89 65 A1 B2 00	101	65 43 21 98 76 65 43 32
10:	100	87 54 00 89 65 A1 B2 00	101	00 11 22 33 44 55 66 77
11:	101	54 A7 00 91 23 66 32 11	111	00 11 22 33 44 55 66 77
	3	64	3	64

Parallel!   ?=   Match

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   23

Set Associative Mapping

- Set associative cache with set size k=2
  = 2-way cache (common)
- Degree of associativity = nr of blocks in a set = v
  - Large degree of associativity?
    - More data items in one set
    - Less "collisions" within set
    - Final comparison (matching tags?) gate delay?
  - Maximum (nr of cache lines)
    ⇒ fully associative mapping   Whole cache is one set!
  - Minimum (1)
    ⇒ direct mapping   Each cache line is a set!

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   24

Cache Replacement Algorithm

- Which cache block to replace
  to make room for new block from memory?
- Direct mapping: trivial
- First-In-First-Out (FIFO)?
- Least-Frequently-Used (LFU)?
- Random?
- Which one is best / possible?
  - Chip area?
  - Fast? Easy to implement?

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   25

Cache Write Policy – memory writes?

- Write through (*läpikirjoittava*)
  - Each write goes always to cache and memory
  - Each write is a cache miss!
- Write back (*lopuksi/takaisin kirjoittava*)
  - Each write goes only to cache
  - Write cache block back to memory
  - only when it is replaced in cache   A bit set
  - Memory may have stale (old) data
  - cache coherence problem (*eheys, yhdenmukaisuus*)
- Write once (*"vain kerran kirjoittava?"*)
  - Write-invalidate Snoopy-cache coherence protocol for multiprocessors
  - Write invalidates data in other caches
  - Write to memory at replacement time, or when some other cache needs it (has read/write miss)   Discussion?

Coherence problems:
More users of the same data:
memory valid?
cache valid?
multiple processors with own caches

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   26

Cache Line Size

- How big cache line?
- Optimise for temporal or spatial locality?
  - bigger cache line
    ⇒ better for spatial locality
  - more cache lines
    ⇒ better for temporal locality
- Best size varies with program or program phase?
- Best size different with code and data?
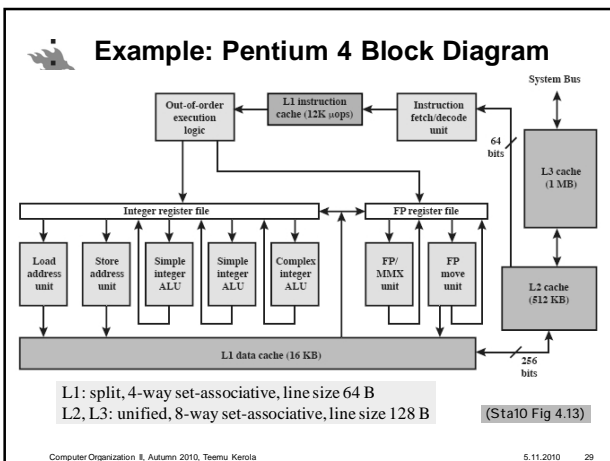- 2-8 words?
  - word = 1 float??

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   27

Types and Number of Caches

- Same cache for <u>data</u> and <u>code</u>, or not?
  - Data references and code references behave differently
  - Unified vs. split cache (*yhdistetty/erilliset*)
  - Split cache: can optimise structure separately for data and code
    Trend towards split caches: Pentium, Power PC, ARM..
- One cache too large for best results?
  - "Smaller is faster"
- Multiple levels of caches
  - L1 on same chip as CPU
  - L2 on same package or chip as CPU
    - older systems: same board
  - L3 on same board as CPU

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   28

Example: Pentium 4 Block Diagram

L1: split, 4-way set-associative, line size 64 B
L2, L3: unified, 8-way set-associative, line size 128 B   (Sta10 Fig 4.13)

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   29

Computer Organization II

Main Memory

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   30

Main Memory Types

(katoava, haihtuva)

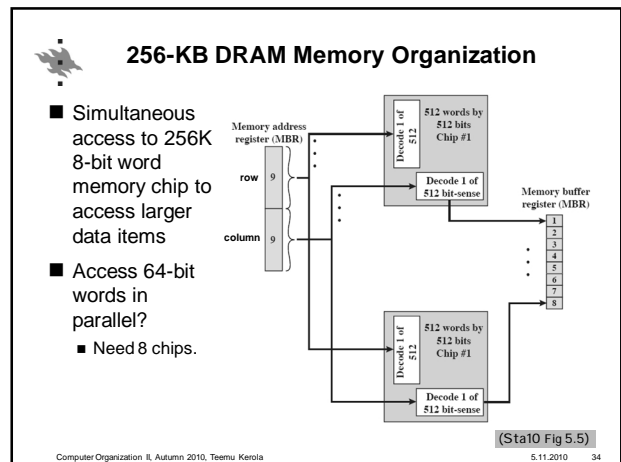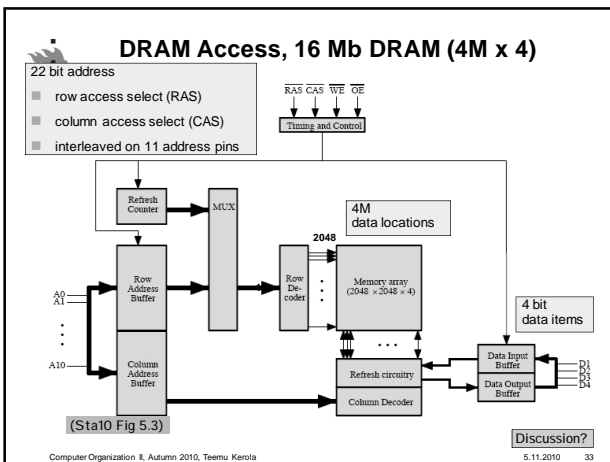| Memory Type	Category	Erasure	Write Mechanism	Volatility
Random-access memory (RAM)	Read-write memory	Electrically, byte-level	Electrically	Volatile
Read-only memory (ROM)	Read-only memory	Not possible	Masks	Nonvolatile
Programmable ROM (PROM)				
Erasable PROM (EPROM)	Read-mostly memory	UV light, chip-level	Electrically	
Electrically Erasable PROM (EEPROM)		Electrically, byte-level		
Flash memory		Electrically, block-level		

(Sta10 Table 5.1)

- **Random access semiconductor memory**
  - Direct access to each memory cell
  - Access time same for all cells

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   31

RAM

- **Dynamic RAM, DRAM**
  - Periodic refreshing required
  - Refresh required after read
  - Simpler, slower, denser, bigger (bytes per chip)
  - Access time ~ 60 ns (?)
  - Main memory? (early systems)

  Analog: Charge on capacitors

- **Static RAM, SRAM**
  - No periodic refreshing needed
  - Data remains until power is lost
  - More complex (more chip area/byte), faster, smaller
  - Access time ~ 25 ns (?)
  - Cache?

  Digital: flip-flop gates

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   32

DRAM Access, 16 Mb DRAM (4M x 4)

22 bit address
- row access select (RAS)
- column access select (CAS)
- interleaved on 11 address pins

(Sta10 Fig 5.3)

Discussion?

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   33

256-KB DRAM Memory Organization

- Simultaneous access to 256K 8-bit word memory chip to access larger data items
- Access 64-bit words in parallel?
  - Need 8 chips.
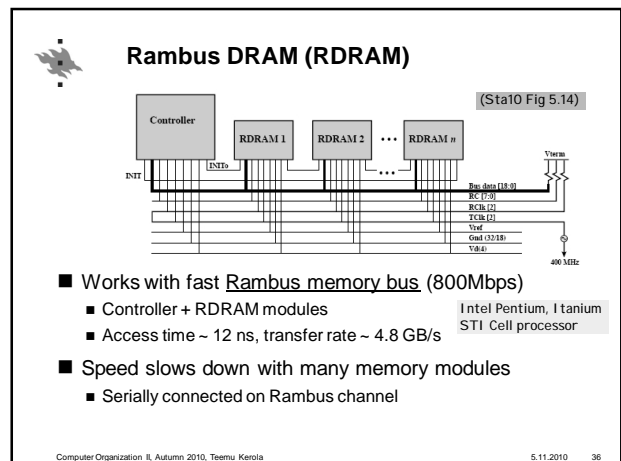

(Sta10 Fig 5.5)

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   34

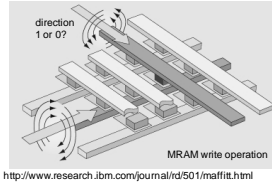## SDRAM (Synchronous DRAM)

- **CPU clock synchronizes also the bus**
  - Runs on higher clock speeds than ordinary DRAM
  - CPU knows how long it takes to make a reference,
    - can do other work while waiting
- **16 bits in parallel**
  - Access 4 DRAMs (4 bits each) in parallel
  - Access time ~ 18 ns, transfer rate ~ 1.3 GB/s
- **DDR SDRAM, double data rate**
  - Current main memory technology
  - Supports transfers both on rising and falling edge of the clock cycle
  - Consumes less power
  - Access time ~ 12 ns, transfer rate ~ 3.2 GB/s

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   35

Rambus DRAM (RDRAM)

(Sta10 Fig 5.14)

- **Works with fast Rambus memory bus (800Mbps)**
  - Controller + RDRAM modules
  - Access time ~ 12 ns, transfer rate ~ 4.8 GB/s

  Intel Pentium, Itanium STI Cell processor

- **Speed slows down with many memory modules**
  - Serially connected on Rambus channel

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   36

MRAM

- Magnetoresistive Random Access Memory (MRAM)
  - Data stored with magnetic fields on two plates
  - Magnetic field directions determine bit value
- Non-volatile, data remains with power off
  - Fast to read/write
  - No upper limit for write counts (Flash has upper limit)
  - Access time comparable to DRAM, almost as fast as SRAM
- Future open
  - Small market share now
  - Expensive now (2006: $25 4Mbit, 2008: $15 4Mbit, Freescale)
  - Still under development
  - May replace flash "*in a few years*"
    - Has not by 2010 …

direction 1 or 0?

MRAM write operation

http://www.research.ibm.com/journal/rd/501/maffitt.html

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   37

Summary

- Memory hierarchy
- Cache
  - Size, Line size, Mapping, nr of levels, unified or split, write policy, replacement policy
- Memory
  - DRAM, SRAM
  - Overall features, no details required
  - Current technologies
    - Synchronous DRAM (SDRAM)
    - Double Data Rate DRAM (DDR)
    - Rambus DRAM

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   383838

Review questions

- Memory hierarchy and principle of locality?
- Different ways to use locality in cache solutions?
- Differences of associative and set associative mappings?
- Why to have separate caches for instructions and data ?

Computer Organization II, Autumn 2010, Teemu Kerola   5.11.2010   39